

Foundation of Cryptography
(0368-4162-01), Lecture 1
Adminstration + Introduction

Iftach Haitner, Tel Aviv University

November first, 2011

Part I

Administration and Course Overview

Section 1

Administration

Important Details

- 1 Iftach Haitner. Schriber 20, email iftachh at gmail.com
- 2 Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- 3 Who are you?
- 4 Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- 5 Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Important Details

- ❶ Iftach Haitner. Schriber 20, email iftachh at gmail.com
- ❷ Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- ❸ Who are you?
- ❹ Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- ❺ Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Important Details

- ❶ Iftach Haitner. Schriber 20, email iftachh at gmail.com
- ❷ Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- ❸ Who are you?
- ❹ Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- ❺ Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Important Details

- ❶ Iftach Haitner. Schriber 20, email iftachh at gmail.com
- ❷ Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- ❸ Who are you?
- ❹ Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- ❺ Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Important Details

- ❶ Iftach Haitner. Schriber 20, email iftachh at gmail.com
- ❷ Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- ❸ Who are you?
- ❹ Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- ❺ Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Important Details

- ❶ Iftach Haitner. Schriber 20, email iftachh at gmail.com
- ❷ Reception: Sundays 9:00-10:00 (please coordinate via email in advance)
- ❸ Who are you?
- ❹ Mailing list: 0368-4162-01@listserv.tau.ac.il
 - Registered students are automatically on the list (need to activate the account by going to <https://www.tau.ac.il/newuser/>)
 - If you're not registered and want to get on the list (or want to get another address on the list), send e-mail to: listserv@listserv.tau.ac.il with the line: subscribe 0368-3500-34 <Real Name>
- ❺ Course website:
<http://www.cs.tau.ac.il/~iftachh/Courses/FOC/Fall11/main.html> (or just Google iftach and follow the link)

Grades

- 1 Grading: Please add your name and email through the course website
 - 1 Class exam 60%
 - 2 Homework 30%: 3-5 exercises. Recommend to use LaTeX (see link in course website) Exercises (separate email per question) should be sent to foc.exc@gmail.com; Title: Question #, Name, Id
 - 3 Self grading 10 %
 - Please register following the link on the course website, and email foc.exc@gmail.com; Title: Grader #: Name, ID
 - Submit your solution to the question using Latex (I'll check it)
 - Within two weeks after the submission time. The grader should send the checked exercises to foc.exc@gmail.com and to the authors, and send a single excel file (columns: Id, Name, grade) to foc.exc@gmail.com, Title: Checked Exe # ,

Grades

- 1 Grading: Please add your name and email through the course website
 - 1 Class exam 60%
 - 2 Homework 30%: 3-5 exercises. Recommend to use LaTeX (see link in course website) Exercises (separate email per question) should be sent to foc.exc@gmail.com; Title: Question #, Name, Id
 - 3 Self grading 10 %
 - Please register following the link on the course website, and email foc.exc@gmail.com; Title: Grader #: Name, ID
 - Submit your solution to the question using Latex (I'll check it)
 - Within two weeks after the submission time. The grader should send the checked exercises to foc.exc@gmail.com and to the authors, and send a single excel file (columns: Id, Name, grade) to foc.exc@gmail.com, Title: Checked Exe # ,

Grades

- ① Grading: Please add your name and email through the course website
 - ① Class exam 60%
 - ② Homework 30%: 3-5 exercises. Recommend to use LaTeX (see link in course website) Exercises (separate email per question) should be sent to foc.exc@gmail.com; Title: Question #, Name, Id
 - ③ Self grading 10 %
 - Please register following the link on the course website, and email foc.exc@gmail.com; Title: Grader #: Name, ID
 - Submit your solution to the question using Latex (I'll check it)
 - Within two weeks after the submission time. The grader should send the checked exercises to foc.exc@gmail.com and to the authors, and send a single excel file (columns: Id, Name, grade) to foc.exc@gmail.com, Title: Checked Exe # ,

and..

1 Slides

2 English

and..

- 1 Slides
- 2 English

Course Prerequisites

- 1 Some prior knowledge of cryptography (such as 0369.3049) might help, but not necessarily
- 2 Basic probability.
- 3 Basic complexity (the classes P, NP, BPP)

Course Material

1 Books:

- 1 Oded Goldreich. Foundations of Cryptography.
- 2 Jonathan Katz and Yehuda Lindell. An Introduction to Modern Cryptography.

2 Lecture notes

- 1 Ran Canetti. Foundation of Cryptography (The 2008 course)
- 2 Salil Vadhan. Introduction to Cryptography.
- 3 Luca Trevisan. Cryptography.
- 4 Yehuda lindell Foundations of Cryptography.

Section 2

Course Topics

Course Topics

Basic primitives in cryptography (i.e., one-way functions, pseudorandom generators and zero-knowledge proofs).

- Focus on *formal* definitions and *rigorous* proofs.
- The goal is not studying some list, but to understand cryptography.
- Get ready to start researching

Part II

Foundation of Cryptography

Cryptography and Computational Hardness

1 What is Cryptography?

2 Hardness assumptions, why do we need them?

3 Does $P \neq NP$ suffice?

$P \neq NP$: i.e., $\exists L \in NP$, such that for any polynomial-time algorithm A , $\exists x \in \{0, 1\}^*$ with $A(x) \neq 1_L(x)$

polynomial-time algorithms: an algorithm A runs in polynomial-time, if $\exists p \in \text{poly}$ such that the running time of $A(x)$ is bounded by $p(|x|)$ for any $x \in \{0, 1\}^*$

4 Problems: hard on the average. No known solution

5 One-way functions: an efficiently computable function that no efficient algorithm can invert.

Cryptography and Computational Hardness

- 1 What is Cryptography?
- 2 Hardness assumptions, why do we need them?
- 3 Does $P \neq NP$ suffice?

$P \neq NP$: i.e., $\exists L \in NP$, such that for any polynomial-time algorithm A , $\exists x \in \{0, 1\}^*$ with $A(x) \neq 1_L(x)$

polynomial-time algorithms: an algorithm A runs in polynomial-time, if $\exists p \in \text{poly}$ such that the running time of $A(x)$ is bounded by $p(|x|)$ for any $x \in \{0, 1\}^*$

- 4 Problems: hard on the average. No known solution
- 5 One-way functions: an efficiently computable function that no efficient algorithm can invert.

Cryptography and Computational Hardness

- 1 What is Cryptography?
- 2 Hardness assumptions, why do we need them?
- 3 Does $P \neq NP$ suffice?

$P \neq NP$: i.e., $\exists L \in NP$, such that for any polynomial-time algorithm A , $\exists x \in \{0, 1\}^*$ with $A(x) \neq 1_L(x)$

polynomial-time algorithms: an algorithm A runs in polynomial-time, if $\exists p \in \text{poly}$ such that the running time of $A(x)$ is bounded by $p(|x|)$ for any $x \in \{0, 1\}^*$

- 1 Problems: hard on the average. No known solution
- 2 One-way functions: an efficiently computable function that no efficient algorithm can invert.

Cryptography and Computational Hardness

- 1 What is Cryptography?
- 2 Hardness assumptions, why do we need them?
- 3 Does $P \neq NP$ suffice?

$P \neq NP$: i.e., $\exists L \in NP$, such that for any polynomial-time algorithm A , $\exists x \in \{0, 1\}^*$ with $A(x) \neq 1_L(x)$

polynomial-time algorithms: an algorithm A runs in polynomial-time, if $\exists p \in \text{poly}$ such that the running time of $A(x)$ is bounded by $p(|x|)$ for any $x \in \{0, 1\}^*$

- 4 Problems: hard on the average. No known solution
- 5 One-way functions: an efficiently computable function that no efficient algorithm can invert.

Cryptography and Computational Hardness

- ❶ What is Cryptography?
- ❷ Hardness assumptions, why do we need them?
- ❸ Does $P \neq NP$ suffice?
 - $P \neq NP$:** i.e., $\exists L \in NP$, such that for any polynomial-time algorithm A , $\exists x \in \{0, 1\}^*$ with $A(x) \neq 1_L(x)$
 - polynomial-time algorithms:** an algorithm A runs in polynomial-time, if $\exists p \in \text{poly}$ such that the running time of $A(x)$ is bounded by $p(|x|)$ for any $x \in \{0, 1\}^*$
- ❹ Problems: hard on the average. No known solution
- ❺ One-way functions: an efficiently computable function that no efficient algorithm can invert.

Section 4

One Way Functions

One-Way Functions

Definition 1 (One-Way Functions (OWFs))

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is one-way, if for any PPT A

$$\Pr_{y \leftarrow f(U_n)}[A(1^n, y) \in f^{-1}(y)] = \text{neg}(n)$$

U_n : a random variable uniformly distributed over $\{0, 1\}^n$

polynomial-time computable: there exists a polynomial-time algorithm F , such that $F(x) = f(x)$ for every $x \in \{0, 1\}^*$

PPT: probabilistic polynomial-time algorithm

neg: a function $g: \mathbb{N} \mapsto [0, 1]$ is a *negligible* function of n , denoted $g(n) = \text{neg}(n)$, if for any $p \in \text{poly}$ there exists $n' \in \mathbb{N}$ such that $g(n) < 1/p(n)$ for all $n > n'$

We will typically omit 1^n from the parameter list of A

- ❶ Is this the right definition?
 - Asymptotic
 - Efficiently computable
 - On the average
 - Only against PPT's
- ❷ (most) Crypto implies OWFs
- ❸ Do OWFs imply Crypto?
- ❹ Where do we find them
- ❺ Non uniform OWFs

Definition 2 (Non-uniform OWF))

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is one-way, if for any polynomial-size family of circuits $\{C_n\}_{n \in \mathbb{N}}$

$$\Pr_{y \leftarrow f(U_n)}[C_n(y) \in f^{-1}(y)] = \text{neg}(n)$$

- ❶ Is this the right definition?
 - Asymptotic
 - Efficiently computable
 - On the average
 - Only against PPT's
- ❷ (most) Crypto implies OWFs
- ❸ Do OWFs imply Crypto?
- ❹ Where do we find them
- ❺ Non uniform OWFs

Definition 2 (Non-uniform OWF))

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is one-way, if for any polynomial-size family of circuits $\{C_n\}_{n \in \mathbb{N}}$

$$\Pr_{y \leftarrow f(U_n)}[C_n(y) \in f^{-1}(y)] = \text{neg}(n)$$

- ❶ Is this the right definition?
 - Asymptotic
 - Efficiently computable
 - On the average
 - Only against PPT's
- ❷ (most) Crypto implies OWFs
- ❸ Do OWFs imply Crypto?
- ❹ Where do we find them
- ❺ Non uniform OWFs

Definition 2 (Non-uniform OWF))

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is one-way, if for any polynomial-size family of circuits $\{C_n\}_{n \in \mathbb{N}}$

$$\Pr_{y \leftarrow f(U_n)}[C_n(y) \in f^{-1}(y)] = \text{neg}(n)$$

Length preserving functions

Definition 3 (length preserving functions)

A function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is length preserving, if $|f(x)| = |x|$ for any $x \in \{0, 1\}^*$

Theorem 4

Assume that OWFs exist, then there exist length-preserving OWFs

Proof idea: use the assumed OWF to create a length preserving one

Length preserving functions

Definition 3 (length preserving functions)

A function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is length preserving, if $|f(x)| = |x|$ for any $x \in \{0, 1\}^*$

Theorem 4

Assume that OWFs exist, then there exist length-preserving OWFs

Proof idea: use the assumed OWF to create a length preserving one

Length preserving functions

Definition 3 (length preserving functions)

A function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is length preserving, if $|f(x)| = |x|$ for any $x \in \{0, 1\}^*$

Theorem 4

Assume that OWFs exist, then there exist length-preserving OWFs

Proof idea: use the assumed OWF to create a length preserving one

OWFs imply Length Preserving OWFs

Definition 5 (Partial domain functions)

For $m, \ell: \mathbb{N} \mapsto \mathbb{N}$, let $h: \{0, 1\}^{m(n)} \mapsto \{0, 1\}^{\ell(n)}$ denote a function defined over input lengths in $\{m(n)\}_{n \in \mathbb{N}}$, and maps strings of length $m(n)$ to strings of length $\ell(n)$. The definition of one-wayness naturally extends to such functions.

Let f be a OWF, let $p \in \text{poly}$ be a bound on its computing-time and assume wlg. that p is monotonly increasing (can we?).

Construction 6 (the length preserving function)

Define $g: \{0, 1\}^{p(n)} \mapsto \{0, 1\}^{p(n)}$ as

$$g(x) = f(x_1, \dots, x_n), 0^{p(n) - |f(x_1, \dots, x_n)|}$$

Note that g is length preserving and efficient (why?).

Claim 7

OWFs imply Length Preserving OWFs

Definition 5 (Partial domain functions)

For $m, \ell: \mathbb{N} \mapsto \mathbb{N}$, let $h: \{0, 1\}^{m(n)} \mapsto \{0, 1\}^{\ell(n)}$ denote a function defined over input lengths in $\{m(n)\}_{n \in \mathbb{N}}$, and maps strings of length $m(n)$ to strings of length $\ell(n)$. The definition of one-wayness naturally extends to such functions.

Let f be a OWF, let $p \in \text{poly}$ be a bound on its computing-time and assume wlg. that p is monotonly increasing (can we?).

Construction 6 (the length preserving function)

Define $g: \{0, 1\}^{p(n)} \mapsto \{0, 1\}^{p(n)}$ as

$$g(x) = f(x_{1,\dots,n}), 0^{p(n)-|f(x_{1,\dots,n})|}$$

Note that g is length preserving and efficient (why?).

OWFs imply Length Preserving OWFs

Definition 5 (Partial domain functions)

For $m, \ell: \mathbb{N} \mapsto \mathbb{N}$, let $h: \{0, 1\}^{m(n)} \mapsto \{0, 1\}^{\ell(n)}$ denote a function defined over input lengths in $\{m(n)\}_{n \in \mathbb{N}}$, and maps strings of length $m(n)$ to strings of length $\ell(n)$. The definition of one-wayness naturally extends to such functions.

Let f be a OWF, let $p \in \text{poly}$ be a bound on its computing-time and assume wlg. that p is monotonly increasing (can we?).

Construction 6 (the length preserving function)

Define $g: \{0, 1\}^{p(n)} \mapsto \{0, 1\}^{p(n)}$ as

$$g(x) = f(x_1, \dots, x_n), 0^{p(n) - |f(x_1, \dots, x_n)|}$$

Note that g is length preserving and efficient (why?).

Claim 7

Proving that g is one-way

How can we prove that g is one-way?

Answer: using reduction

Proof:

Assume that g is not one-way. Namely, there exists PPT A a $q \in \text{poly}$ and an infinite $\mathcal{I} \subseteq \{p(n) : n \in \mathbb{N}\}$ such that

$$\Pr_{y \leftarrow g(U_n)}[A(y) \in g^{-1}(y)] > 1/q(n) \quad (1)$$

for any $n \in \mathcal{I}$ (?).

We would like to use A for inverting f .

Proving that g is one-way

How can we prove that g is one-way?

Answer: using reduction

Proof:

Assume that g is not one-way. Namely, there exists PPT A a $q \in \text{poly}$ and an infinite $\mathcal{I} \subseteq \{p(n) : n \in \mathbb{N}\}$ such that

$$\Pr_{y \leftarrow g(U_n)}[A(y) \in g^{-1}(y)] > 1/q(n) \quad (1)$$

for any $n \in \mathcal{I}$ (?).

We would like to use A for inverting f .

Proving that g is one-way

How can we prove that g is one-way?

Answer: using reduction

Proof:

Assume that g is not one-way. Namely, there exists PPT A a $q \in \text{poly}$ and an infinite $\mathcal{I} \subseteq \{p(n) : n \in \mathbb{N}\}$ such that

$$\Pr_{y \leftarrow g(U_n)}[A(y) \in g^{-1}(y)] > 1/q(n) \quad (1)$$

for any $n \in \mathcal{I}$ (?).

We would like to use A for inverting f .

Proving that g is one-way

How can we prove that g is one-way?

Answer: using reduction

Proof:

Assume that g is not one-way. Namely, there exists PPT A a $q \in \text{poly}$ and an infinite $\mathcal{I} \subseteq \{p(n) : n \in \mathbb{N}\}$ such that

$$\Pr_{y \leftarrow g(U_n)}[A(y) \in g^{-1}(y)] > 1/q(n) \quad (1)$$

for any $n \in \mathcal{I}$ (?).

We would like to use A for inverting f .

Algorithm 8 (The inverter B)

Input: 1^n and $y \in \{0, 1\}^*$.

- 1 Let $x = A(1^{p(n)}, y, 0^{p(n)-|y|})$.
- 2 Return $x_{1,\dots,n}$.

Claim 9

Let $\mathcal{I}' := \{n \in \mathbb{N} : p(n) \in \mathcal{I}\}$. Then

- 1 \mathcal{I}' is infinite
- 2 For any $n \in \mathcal{I}'$, it holds that $\Pr_{y \leftarrow g(u_n)}[B(y) \in f^{-1}(y)] > 1/q(p(n))$.

in contradiction to the assumed one-wayness of f . \square

Algorithm 8 (The inverter B)

Input: 1^n and $y \in \{0, 1\}^*$.

- 1 Let $x = A(1^{p(n)}, y, 0^{p(n)-|y|})$.
- 2 Return $x_{1,\dots,n}$.

Claim 9

Let $\mathcal{I}' := \{n \in \mathbb{N} : p(n) \in \mathcal{I}\}$. Then

- 1 \mathcal{I}' is infinite
- 2 For any $n \in \mathcal{I}'$, it holds that $\Pr_{y \leftarrow g(U_n)}[B(y) \in f^{-1}(y)] > 1/q(p(n))$.

in contradiction to the assumed one-wayness of f . \square

Conclusion

Remark 10

- We directly related the hardness of f to that of g
- The reduction is not “security preserving”

From partial domain functions to all-length functions

Construction 11

Given a function $f: \{0, 1\}^{m(n)} \mapsto \{0, 1\}^{\ell(n)}$,
 $f_{all}: \{0, 1\}^* \mapsto \{0, 1\}^*$ as

$$f_{all}(x) = f(x_1, \dots, x_{k(n)}), 0^{n-k(n)}$$

where $n = |x|$ and $k(n) := \max\{m(n') \leq n: n' \in \mathbb{N}\}$.

Claim 12

Assume that f is a one-way function and that m is monotone, polynomial-time computable and satisfies $\frac{m(n+1)}{m(n)} \leq p(n)$ for some $p \in \text{poly}$, then f_{all} is a one-way function. Further, if f is length preserving, then so is f_{all} .

Proof: ?

From partial domain functions to all-length functions

Construction 11

Given a function $f: \{0, 1\}^{m(n)} \mapsto \{0, 1\}^{\ell(n)}$,
 $f_{all}: \{0, 1\}^* \mapsto \{0, 1\}^*$ as

$$f_{all}(x) = f(x_1, \dots, x_{k(n)}), 0^{n-k(n)}$$

where $n = |x|$ and $k(n) := \max\{m(n') \leq n: n' \in \mathbb{N}\}$.

Claim 12

Assume that f is a one-way function and that m is monotone, polynomial-time computable and satisfies $\frac{m(n+1)}{m(n)} \leq p(n)$ for some $p \in \text{poly}$, then f_{all} is a one-way function. Further, if f is length preserving, then so is f_{all} .

Proof: ?

Weak One Way Functions

Definition 13 (weak one-way functions)

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is α -one-way, if

$$\Pr_{y \leftarrow f(U_n)}[A(1^n, y) \in f^{-1}(y)] \leq \alpha(n)$$

for any PPT A and large enough $n \in \mathbb{N}$.

- 1 (strong) OWF according to Def 1, are $\text{neg}(n)$ -one-way according to the above definition
- 2 Examples
- 3 Can we “amplify” weak OWF to strong ones?

Weak One Way Functions

Definition 13 (weak one-way functions)

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is α -one-way, if

$$\Pr_{y \leftarrow f(U_n)}[A(1^n, y) \in f^{-1}(y)] \leq \alpha(n)$$

for any PPT A and large enough $n \in \mathbb{N}$.

- 1 (strong) OWF according to Def 1, are $\text{neg}(n)$ -one-way according to the above definition
- 2 Examples
- 3 Can we “amplify” weak OWF to strong ones?

Weak One Way Functions

Definition 13 (weak one-way functions)

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is α -one-way, if

$$\Pr_{y \leftarrow f(U_n)}[A(1^n, y) \in f^{-1}(y)] \leq \alpha(n)$$

for any PPT A and large enough $n \in \mathbb{N}$.

- 1 (strong) OWF according to Def 1, are $\text{neg}(n)$ -one-way according to the above definition
- 2 Examples
- 3 Can we “amplify” weak OWF to strong ones?

Weak One Way Functions

Definition 13 (weak one-way functions)

A polynomial-time computable function $f : \{0, 1\}^* \mapsto \{0, 1\}^*$ is α -one-way, if

$$\Pr_{y \leftarrow f(U_n)}[A(1^n, y) \in f^{-1}(y)] \leq \alpha(n)$$

for any PPT A and large enough $n \in \mathbb{N}$.

- 1 (strong) OWF according to Def 1, are $\text{neg}(n)$ -one-way according to the above definition
- 2 Examples
- 3 Can we “amplify” weak OWF to strong ones?

Strong to weak OWFs

Claim 14

Assume there exists OWFs, then there exist functions that are $\frac{1}{3}$ one-way, but not (strong) one-way

Proof: let f be a owf. Define $g(x) = (1, g(x))$ if $x_1 = 1$, and 0 otherwise.

Strong to weak OWFs

Claim 14

Assume there exists OWFs, then there exist functions that are $\frac{1}{3}$ one-way, but not (strong) one-way

Proof: let f be a owf. Define $g(x) = (1, g(x))$ if $x_1 = 1$, and 0 otherwise.

Weak to Strong OWFs

Theorem 15

Assume there exists $(1 - \alpha)$ -weak OWFs with $\alpha(n) > 1/p(n)$ for some $p \in \text{poly}$, then there exists (strong) one-way functions.

Proof: we assume wlg that f is length preserving (can we do so?)

Construction 16 (g – the strong one-way function)

Let $t: \mathbb{N} \mapsto \mathbb{N}$ be a polynomial-time computable function satisfying $t(n) \in \omega(\log n / \alpha(n))$. Define $g: (\{0, 1\}^n)^{t(n)} \mapsto (\{0, 1\}^n)^{t(n)}$ as

$$g(x_1, \dots, x_t) = f(x_1), \dots, f(x_t)$$

Claim 17

g is one-way.

Weak to Strong OWFs

Theorem 15

Assume there exists $(1 - \alpha)$ -weak OWFs with $\alpha(n) > 1/p(n)$ for some $p \in \text{poly}$, then there exists (strong) one-way functions.

Proof: we assume wlg that f is length preserving (can we do so?)

Construction 16 (g – the strong one-way function)

Let $t: \mathbb{N} \mapsto \mathbb{N}$ be a polynomial-time computable function satisfying $t(n) \in \omega(\log n / \alpha(n))$. Define $g: (\{0, 1\}^n)^{t(n)} \mapsto (\{0, 1\}^n)^{t(n)}$ as

$$g(x_1, \dots, x_t) = f(x_1), \dots, f(x_t)$$

Claim 17

g is one-way.

Weak to Strong OWFs

Theorem 15

Assume there exists $(1 - \alpha)$ -weak OWFs with $\alpha(n) > 1/p(n)$ for some $p \in \text{poly}$, then there exists (strong) one-way functions.

Proof: we assume wlg that f is length preserving (can we do so?)

Construction 16 (g – the strong one-way function)

Let $t: \mathbb{N} \mapsto \mathbb{N}$ be a polynomial-time computable function satisfying $t(n) \in \omega(\log n / \alpha(n))$. Define $g: (\{0, 1\}^n)^{t(n)} \mapsto (\{0, 1\}^n)^{t(n)}$ as

$$g(x_1, \dots, x_t) = f(x_1), \dots, f(x_t)$$

Claim 17

g is one-way.

Proving that g is one-way – the naive approach

Let A be a potential inverter for g , and assume that A tries to attacks each of the t outputs of g *independently*. Then

$$\Pr_{y \leftarrow g(U_n^{t(n)})} [A(y) \in g^{-1}(y)] \leq (1 - \alpha(n))^{t(n)} \leq e^{-\omega(\log n)} = \text{neg}(n)$$

A less naive approach would be to assume that A goes over output sequentially.

Unfortunately, we can assume none of the above.

Proving that g is one-way – the naive approach

Let A be a potential inverter for g , and assume that A tries to attacks each of the t outputs of g *independently*. Then

$$\Pr_{y \leftarrow g(U_n^{t(n)})} [A(y) \in g^{-1}(y)] \leq (1 - \alpha(n))^{t(n)} \leq e^{-\omega(\log n)} = \text{neg}(n)$$

A less naive approach would be to assume that A goes over output sequentially.

Unfortunately, we can assume none of the above.

Proving that g is one-way – the naive approach

Let A be a potential inverter for g , and assume that A tries to attacks each of the t outputs of g *independently*. Then

$$\Pr_{y \leftarrow g(U_n^{t(n)})} [A(y) \in g^{-1}(y)] \leq (1 - \alpha(n))^{t(n)} \leq e^{-\omega(\log n)} = \text{neg}(n)$$

A less naive approach would be to assume that A goes over output sequentially.

Unfortunately, we can assume none of the above.

Failing Sets

Definition 18 (failing set)

A function $f : \{0, 1\}^n \mapsto \{0, 1\}^{\ell(n)}$ has a $(\delta(n), \varepsilon(n))$ -failing set for A , if for large enough n , exists set $S(n) \subseteq \{0, 1\}^{\ell(n)}$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq \delta(n)$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] < \varepsilon(n)$, for every $y \in S(n)$

Claim 19

Let f be a $(1 - \alpha)$ -OWF. Then f has $(\alpha(n)/2, 1/p(n))$ -failing set for any PPT A and $p \in \text{poly}$.

Proof: Assume \exists PPT A , a $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ such that for every $n \in \mathcal{I}$, $\exists S(n) \subseteq \{0, 1\}^n$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq 1 - \alpha(n)/2$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] \geq 1/p(n)$, for every $y \in S(n)$

We'll use A to contradict the hardness of f .

Failing Sets

Definition 18 (failing set)

A function $f : \{0, 1\}^n \mapsto \{0, 1\}^{\ell(n)}$ has a $(\delta(n), \varepsilon(n))$ -failing set for A , if for large enough n , exists set $S(n) \subseteq \{0, 1\}^{\ell(n)}$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq \delta(n)$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] < \varepsilon(n)$, for every $y \in S(n)$

Claim 19

Let f be a $(1 - \alpha)$ -OWF. Then f has $(\alpha(n)/2, 1/p(n))$ -failing set for any PPT A and $p \in \text{poly}$.

Proof: Assume \exists PPT A , a $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ such that for every $n \in \mathcal{I}$, $\exists S(n) \subseteq \{0, 1\}^n$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq 1 - \alpha(n)/2$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] \geq 1/p(n)$, for every $y \in S(n)$

We'll use A to contradict the hardness of f .

Failing Sets

Definition 18 (failing set)

A function $f : \{0, 1\}^n \mapsto \{0, 1\}^{\ell(n)}$ has a $(\delta(n), \varepsilon(n))$ -failing set for A , if for large enough n , exists set $S(n) \subseteq \{0, 1\}^{\ell(n)}$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq \delta(n)$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] < \varepsilon(n)$, for every $y \in S(n)$

Claim 19

Let f be a $(1 - \alpha)$ -OWF. Then f has $(\alpha(n)/2, 1/p(n))$ -failing set for any PPT A and $p \in \text{poly}$.

Proof: Assume \exists PPT A , a $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ such that for every $n \in \mathcal{I}$, $\exists S(n) \subseteq \{0, 1\}^n$ with

- 1 $\Pr[f(U_n) \in S(n)] \geq 1 - \alpha(n)/2$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] \geq 1/p(n)$, for every $y \in S(n)$

We'll use A to contradict the hardness of f .

Failing Sets

Definition 18 (failing set)

A function $f : \{0, 1\}^n \mapsto \{0, 1\}^{\ell(n)}$ has a $(\delta(n), \varepsilon(n))$ -failing set for A , if for large enough n , exists set $\mathcal{S}(n) \subseteq \{0, 1\}^{\ell(n)}$ with

- 1 $\Pr[f(U_n) \in \mathcal{S}(n)] \geq \delta(n)$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] < \varepsilon(n)$, for every $y \in \mathcal{S}(n)$

Claim 19

Let f be a $(1 - \alpha)$ -OWF. Then f has $(\alpha(n)/2, 1/p(n))$ -failing set for any PPT A and $p \in \text{poly}$.

Proof: Assume \exists PPT A , a $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ such that for every $n \in \mathcal{I}$, $\exists \mathcal{S}(n) \subseteq \{0, 1\}^n$ with

- 1 $\Pr[f(U_n) \in \mathcal{S}(n)] \geq 1 - \alpha(n)/2$, and
- 2 $\Pr[A(y) \in f^{-1}(y)] \geq 1/p(n)$, for every $y \in \mathcal{S}(n)$

We'll use A to contradict the hardness of f .

Using A to invert f

Algorithm 20 (The inverter B)

Input: $y \in \{0, 1\}^n$.

Do (with fresh randomness) for $np(n)$ times:

If $x = A(y) \in f^{-1}(y)$, return x

Clearly, B is a PPT

Claim 21

For every $n \in \mathcal{I}$, it holds that

$$\Pr_{y \leftarrow f(U_n)}[B(y) \in f^{-1}(y)] > 1 - \alpha(n)$$

Hence, f is not $(1 - \alpha(n))$ -one-way \square

Using A to invert f

Algorithm 20 (The inverter B)

Input: $y \in \{0, 1\}^n$.

Do (with fresh randomness) for $np(n)$ times:

If $x = A(y) \in f^{-1}(y)$, return x

Clearly, B is a PPT

Claim 21

For every $n \in \mathcal{I}$, it holds that

$$\Pr_{y \leftarrow f(U_n)}[B(y) \in f^{-1}(y)] > 1 - \alpha(n)$$

Hence, f is not $(1 - \alpha(n))$ -one-way \square

Using A to invert f

Algorithm 20 (The inverter B)

Input: $y \in \{0, 1\}^n$.

Do (with fresh randomness) for $np(n)$ times:

If $x = A(y) \in f^{-1}(y)$, return x

Clearly, B is a PPT

Claim 21

For every $n \in \mathcal{I}$, it holds that

$$\Pr_{y \leftarrow f(U_n)}[B(y) \in f^{-1}(y)] > 1 - \alpha(n)$$

Hence, f is not $(1 - \alpha(n))$ -one-way \square

Proof of Claim 21(all probabilities below are also over $y \leftarrow f(U_n)$):

$$\begin{aligned} & \Pr[B(y) \in f^{-1}(y)] \\ & \geq \Pr[B(y) \in f^{-1}(y) \wedge y \in \mathcal{S}(n)] \\ & = \Pr[y \in \mathcal{S}(n)] \cdot \Pr[B(y) \in f^{-1}(y) \mid y \in \mathcal{S}(n)] \\ & \geq (1 - \alpha(n)/2) \cdot (1 - (1 - 1/p(n))^{np(n)}) \\ & \geq (1 - \alpha(n)/2) \cdot (1 - 2^{-n}) > 1 - \alpha(n). \square \end{aligned}$$

Proof of Claim 21(all probabilities below are also over $y \leftarrow f(U_n)$):

$$\begin{aligned} & \Pr[B(y) \in f^{-1}(y)] \\ & \geq \Pr[B(y) \in f^{-1}(y) \wedge y \in \mathcal{S}(n)] \\ & = \Pr[y \in \mathcal{S}(n)] \cdot \Pr[B(y) \in f^{-1}(y) \mid y \in \mathcal{S}(n)] \\ & \geq (1 - \alpha(n)/2) \cdot (1 - (1 - 1/p(n))^{np(n)}) \\ & \geq (1 - \alpha(n)/2) \cdot (1 - 2^{-n}) > 1 - \alpha(n). \square \end{aligned}$$

Proof of Claim 21(all probabilities below are also over $y \leftarrow f(U_n)$):

$$\begin{aligned} & \Pr[B(y) \in f^{-1}(y)] \\ & \geq \Pr[B(y) \in f^{-1}(y) \wedge y \in \mathcal{S}(n)] \\ & = \Pr[y \in \mathcal{S}(n)] \cdot \Pr[B(y) \in f^{-1}(y) \mid y \in \mathcal{S}(n)] \\ & \geq (1 - \alpha(n)/2) \cdot (1 - (1 - 1/p(n))^{np(n)}) \\ & \geq (1 - \alpha(n)/2) \cdot (1 - 2^{-n}) > 1 - \alpha(n). \square \end{aligned}$$

Proof of Claim 21(all probabilities below are also over $y \leftarrow f(U_n)$):

$$\begin{aligned} & \Pr[B(y) \in f^{-1}(y)] \\ & \geq \Pr[B(y) \in f^{-1}(y) \wedge y \in \mathcal{S}(n)] \\ & = \Pr[y \in \mathcal{S}(n)] \cdot \Pr[B(y) \in f^{-1}(y) \mid y \in \mathcal{S}(n)] \\ & \geq (1 - \alpha(n)/2) \cdot (1 - (1 - 1/p(n))^{np(n)}) \\ & \geq (1 - \alpha(n)/2) \cdot (1 - 2^{-n}) > 1 - \alpha(n). \square \end{aligned}$$

Proof of Claim 21(all probabilities below are also over $y \leftarrow f(U_n)$):

$$\begin{aligned} & \Pr[B(y) \in f^{-1}(y)] \\ & \geq \Pr[B(y) \in f^{-1}(y) \wedge y \in \mathcal{S}(n)] \\ & = \Pr[y \in \mathcal{S}(n)] \cdot \Pr[B(y) \in f^{-1}(y) \mid y \in \mathcal{S}(n)] \\ & \geq (1 - \alpha(n)/2) \cdot (1 - (1 - 1/p(n))^{np(n)}) \\ & \geq (1 - \alpha(n)/2) \cdot (1 - 2^{-n}) > 1 - \alpha(n). \square \end{aligned}$$

Proving that g is one-way

We show that if g is not OWF, then f has no flailing-set of the "right" type.

Claim 22

Assume \exists PPT A , $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ s.t.

$$\Pr_{z \leftarrow g(U_n^{t(n)})}[A(z) \in g^{-1}(z)] \geq 1/p(n) \quad (2)$$

for every $n \in \mathcal{I}$. Then \exists PPT B and $q \in \text{poly}$ s.t.

$$\Pr_{y \leftarrow \mathcal{S}}[B(y) \in f^{-1}(y)] \geq 1/q(n) \quad (3)$$

for every $n \in \mathcal{I}$ and $\mathcal{S} \subseteq \{0, 1\}^n$ with $\Pr_{y \leftarrow f(U_n)}[\mathcal{S}] \geq \alpha(n)/2$.

Namely, f does not have a $(\alpha(n)/2, 1/q(n))$ -failing set.

Proving that g is one-way

We show that if g is not OWF, then f has no flailing-set of the "right" type.

Claim 22

Assume \exists PPT A , $p \in \text{poly}$ and an infinite set $\mathcal{I} \subseteq \mathbb{N}$ s.t.

$$\Pr_{z \leftarrow g(U_n^{t(n)})}[A(z) \in g^{-1}(z)] \geq 1/p(n) \quad (2)$$

for every $n \in \mathcal{I}$. Then \exists PPT B and $q \in \text{poly}$ s.t.

$$\Pr_{y \leftarrow \mathcal{S}}[B(y) \in f^{-1}(y)] \geq 1/q(n) \quad (3)$$

for every $n \in \mathcal{I}$ and $\mathcal{S} \subseteq \{0, 1\}^n$ with $\Pr_{y \leftarrow f(U_n)}[\mathcal{S}] \geq \alpha(n)/2$.

Namely, f does not have a $(\alpha(n)/2, 1/q(n))$ -failing set.

Algorithm B

Algorithm 23 (No failing set algorithm B)

Input: $y \in \{0, 1\}^n$.

- 1 Choose $z = (z_1, \dots, z_t) \leftarrow g(U_n^t)$ and $i \leftarrow [t]$
- 2 Set $z' = (z_1, \dots, z_{i-1}, y, z_{i+1}, \dots, z_t)$
- 3 Return $A(z')_i$

Fix $n \in \mathcal{I}$ and a set $\mathcal{S} \subseteq \{0, 1\}^n$ of the right probability. We analyze B's success probability using the following (inefficient) algorithm B*:

Algorithm B

Algorithm 23 (No failing set algorithm B)

Input: $y \in \{0, 1\}^n$.

- 1 Choose $z = (z_1, \dots, z_t) \leftarrow g(U_n^t)$ and $i \leftarrow [t]$
- 2 Set $z' = (z_1, \dots, z_{i-1}, y, z_{i+1}, \dots, z_t)$
- 3 Return $A(z')_i$

Fix $n \in \mathcal{I}$ and a set $\mathcal{S} \subseteq \{0, 1\}^n$ of the right probability. We analyze B's success probability using the following (inefficient) algorithm B*:

Algorithm B*

Definition 24 (Bad)

For $z \in \text{Im}(g)$ (the image of g), we set $\text{Bad}(z) = 1$ iff $\nexists i \in [t]$ with $z_i \in \mathcal{S}$.

B^* differ from B in the way it chooses z' : in case $\text{Bad}(z) = 1$, it sets $z' = z$. Otherwise, it sets i to an arbitrary index $j \in [t]$ with $z_j \in \mathcal{S}$, and sets z' as B does with respect to this i .

Claim 25

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq 1/2p(n)$$

and therefore $\Pr_{y \leftarrow \mathcal{S}}[B(y) \in f^{-1}(y)] \geq 1/2t(n)p(n). \square$

Algorithm B*

Definition 24 (Bad)

For $z \in \text{Im}(g)$ (the image of g), we set $\text{Bad}(z) = 1$ iff $\nexists i \in [t]$ with $z_i \in \mathcal{S}$.

B^* differ from B in the way it chooses z' : in case $\text{Bad}(z) = 1$, it sets $z' = z$. Otherwise, it sets i to an arbitrary index $j \in [t]$ with $z_j \in \mathcal{S}$, and sets z' as B does with respect to this i .

Claim 25

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq 1/2p(n)$$

and therefore $\Pr_{y \leftarrow \mathcal{S}}[B(y) \in f^{-1}(y)] \geq 1/2t(n)p(n).$ \square

Claim 25 follows from the following two claims,

Claim 26

$$\Pr_{z \leftarrow g(U_n^t)}[\text{Bad}(z)] = \text{neg}(n)$$

Claim 27

Let $Z = g(U_n^t)$ and let Z' be the value of z' induced by a random execution of B^* (on a random y in S).
Then Z and Z' are identically distributed.

The claims imply Claim 25.

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z) \wedge \neg \text{Bad}(z)] \quad (4)$$

$$\begin{aligned} & \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] \\ & \leq \Pr[A(z) \in g^{-1}(Z) \wedge \neg \text{Bad}(z)] + \Pr[\text{Bad}(z)] \end{aligned} \quad (5)$$

It follows that

$$\begin{aligned} \Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] & \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] - \text{neg}(n) \\ & \geq 1/2p(n). \end{aligned}$$



The claims imply Claim 25.

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z) \wedge \neg \text{Bad}(z)] \quad (4)$$

$$\begin{aligned} & \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] \\ & \leq \Pr[A(z) \in g^{-1}(Z) \wedge \neg \text{Bad}(z)] + \Pr[\text{Bad}(z)] \end{aligned} \quad (5)$$

It follows that

$$\begin{aligned} \Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] & \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] - \text{neg}(n) \\ & \geq 1/2p(n). \end{aligned}$$



The claims imply Claim 25.

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z) \wedge \neg \text{Bad}(z)] \quad (4)$$

$$\begin{aligned} & \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] \\ & \leq \Pr[A(z) \in g^{-1}(Z) \wedge \neg \text{Bad}(z)] + \Pr[\text{Bad}(z)] \end{aligned} \quad (5)$$

It follows that

$$\begin{aligned} \Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] & \geq \Pr_{z \leftarrow g(U_h^t)}[A(z) \in g^{-1}(z)] - \text{neg}(n) \\ & \geq 1/2p(n). \end{aligned}$$



The claims imply Claim 25.

$$\Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] \geq \Pr_{z \leftarrow g(U_n^t)}[A(z) \in g^{-1}(z) \wedge \neg \text{Bad}(z)] \quad (4)$$

$$\begin{aligned} & \Pr_{z \leftarrow g(U_n^t)}[A(z) \in g^{-1}(z)] \\ & \leq \Pr[A(z) \in g^{-1}(Z) \wedge \neg \text{Bad}(z)] + \Pr[\text{Bad}(z)] \end{aligned} \quad (5)$$

It follows that

$$\begin{aligned} \Pr_{y \leftarrow \mathcal{S}}[B^*(y) \in f^{-1}(y)] & \geq \Pr_{z \leftarrow g(U_n^t)}[A(z) \in g^{-1}(z)] - \text{neg}(n) \\ & \geq 1/2p(n). \end{aligned}$$



Proof of Claim 26?

Proof of Claim 27: Consider the following process for sampling Z_i :

- 1 Let $\beta = \Pr_{y \leftarrow f(U_n)}[S]$. Set $\ell_i = 1$ wp β and $\ell_i = 0$ otherwise.
- 2 If $\ell_i = 1$, let $y \leftarrow f(U_n) \mid y \in S$. Otherwise, set $y \leftarrow f(U_n) \mid y \notin S$.

It is easy to see that the above process is correct (samples Z correctly).

Now all that B^* does, is repeating Step 2 for one of the i 's with $\ell_i = 1$ (if such exists) \square

Proof of Claim 26?

Proof of Claim 27: Consider the following process for sampling Z_i :

- ➊ Let $\beta = \Pr_{y \leftarrow f(U_n)}[S]$. Set $\ell_i = 1$ wp β and $\ell_i = 0$ otherwise.
- ➋ If $\ell_i = 1$, let $y \leftarrow f(U_n) \mid y \in S$. Otherwise, set $y \leftarrow f(U_n) \mid y \notin S$.

It is easy to see that the above process is correct (samples Z correctly).

Now all that B^* does, is repeating Step 2 for one of the i 's with $\ell_i = 1$ (if such exists) \square

Proof of Claim 26?

Proof of Claim 27: Consider the following process for sampling Z_i :

- ➊ Let $\beta = \Pr_{y \leftarrow f(U_n)}[S]$. Set $\ell_i = 1$ wp β and $\ell_i = 0$ otherwise.
- ➋ If $\ell_i = 1$, let $y \leftarrow f(U_n) \mid y \in S$. Otherwise, set $y \leftarrow f(U_n) \mid y \notin S$.

It is easy to see that the above process is correct (samples Z correctly).

Now all that B^* does, is repeating Step 2 for one of the i 's with $\ell_i = 1$ (if such exists) \square

Conclusion

Remark 28 (hardness amplification via parallel repetition)

- Can we give a more efficient (secure) reduction?
- Similar theorems for other cryptographic primitives (e.g., Captchas, general protocols)?
What properties of the weak OWF have we used in the proof?

Conclusion

Remark 28 (hardness amplification via parallel repetition)

- Can we give a more efficient (secure) reduction?
- Similar theorems for other cryptographic primitives (e.g., Captchas, general protocols)?

What properties of the weak OWF have we used in the proof?

Conclusion

Remark 28 (hardness amplification via parallel repetition)

- Can we give a more efficient (secure) reduction?
- Similar theorems for other cryptographic primitives (e.g., Captchas, general protocols)?

What properties of the weak OWF have we used in the proof?